Week 10 Assignment

Code ▾

Week 10 - Assignment

George Cruz

2020-10-31

Week 10 Assignment

Reproduce and extend Sentiment analysis with tidy data

Hide

```
library(tidytext)
library(janeaustenr)
library(dplyr)
library(stringr)
library(tidyr)
```

Starting analysis

Get Sentiment tables

1. AFINN from Finn Årup Nielsen (http://www2.imm.dtu.dk/pubdb/views/publication_details.php?id=6010)

```
## # A tibble: 2,477 x 2
##    word       value
##    <chr>      <dbl>
##  1 abandon       -2
##  2 abandoned     -2
##  3 abandons      -2
##  4 abducted      -2
##  5 abduction     -2
##  6 abductions    -2
##  7 abhor         -3
##  8 abhorred      -3
##  9 abhorrent     -3
## 10 abhors        -3
## # ... with 2,467 more rows
```

2. bing from Bing Liu and collaborators (https://www.cs.uic.edu/~liub/FBS/sentiment-analysis.html)

```
## # A tibble: 6,786 x 2
##    word        sentiment
##    <chr>       <chr>    
##  1 2-faces     negative 
##  2 abnormal    negative 
##  3 abolish     negative 
##  4 abominable  negative 
##  5 abominably  negative 
##  6 abominate   negative 
##  7 abomination negative 
##  8 abort       negative 
##  9 aborted     negative 
## 10 aborts      negative 
## # ... with 6,776 more rows
```

3. NRC from Saif Mohammad and Peter Turney (http://saifmohammad.com/WebPages/NRC-Emotion-Lexicon.htm).

```
## # A tibble: 13,901 x 2
##    word        sentiment
##    <chr>       <chr>    
##  1 abacus      trust    
##  2 abandon     fear     
##  3 abandon     negative 
##  4 abandon     sadness  
##  5 abandoned   anger    
##  6 abandoned   fear     
##  7 abandoned   negative 
##  8 abandoned   sadness  
##  9 abandonment anger    
## 10 abandonment fear     
## # ... with 13,891 more rows
```

Sentiment analysis with inner join

With data in a tidy format, sentiment analysis can be done as an inner join. This is another of the great successes of viewing text mining as a tidy data analysis task; much as removing stop words is an antijoin operation, performing sentiment analysis is an inner join operation.

Hide

```
tidy_books <- austen_books() %>%
  group_by(book) %>%
  mutate(
    linenumber = row_number(),
    chapter = cumsum(str_detect(text, regex("^chapter [\\divxlc]",
      ignore_case = TRUE
    )))
  ) %>%
  ungroup() %>%
  unnest_tokens(word, text)
```

Performing the sentiment Analysis

Looking for words with *joy* sentiment within our data:

Hide

```
nrc_joy <- get_sentiments("nrc") %>%
  filter(sentiment == "joy")

tidy_books %>%
  filter(book == "Emma") %>%
  inner_join(nrc_joy) %>%
  count(word, sort = TRUE)
```

```
## Joining, by = "word"
```

```
## # A tibble: 303 x 2
##    word        n
##    <chr>   <int>
##  1 good      359
##  2 young     192
##  3 friend    166
##  4 hope      143
##  5 happy     125
##  6 love      117
##  7 deal       92
##  8 found      92
##  9 present    89
## 10 kind       82
## # ... with 293 more rows
```

Looking at the overall sentiment in Jane Austen's books:

Small sections of text may not have enough words in them to get a good estimate of sentiment while really large sections can wash out narrative structure. For these books, using 80 lines works well, but this can vary depending on individual texts, how long the lines were to start with, etc. We then use spread() so that we have negative and positive sentiment in separate columns, and lastly calculate a net sentiment (positive - negative).

Hide

```
jane_austen_sentiment <- tidy_books %>%
  inner_join(get_sentiments("bing")) %>%
  count(book, index = linenumber %/% 80, sentiment) %>%
  spread(sentiment, n, fill = 0) %>%
  mutate(sentiment = positive - negative)
```

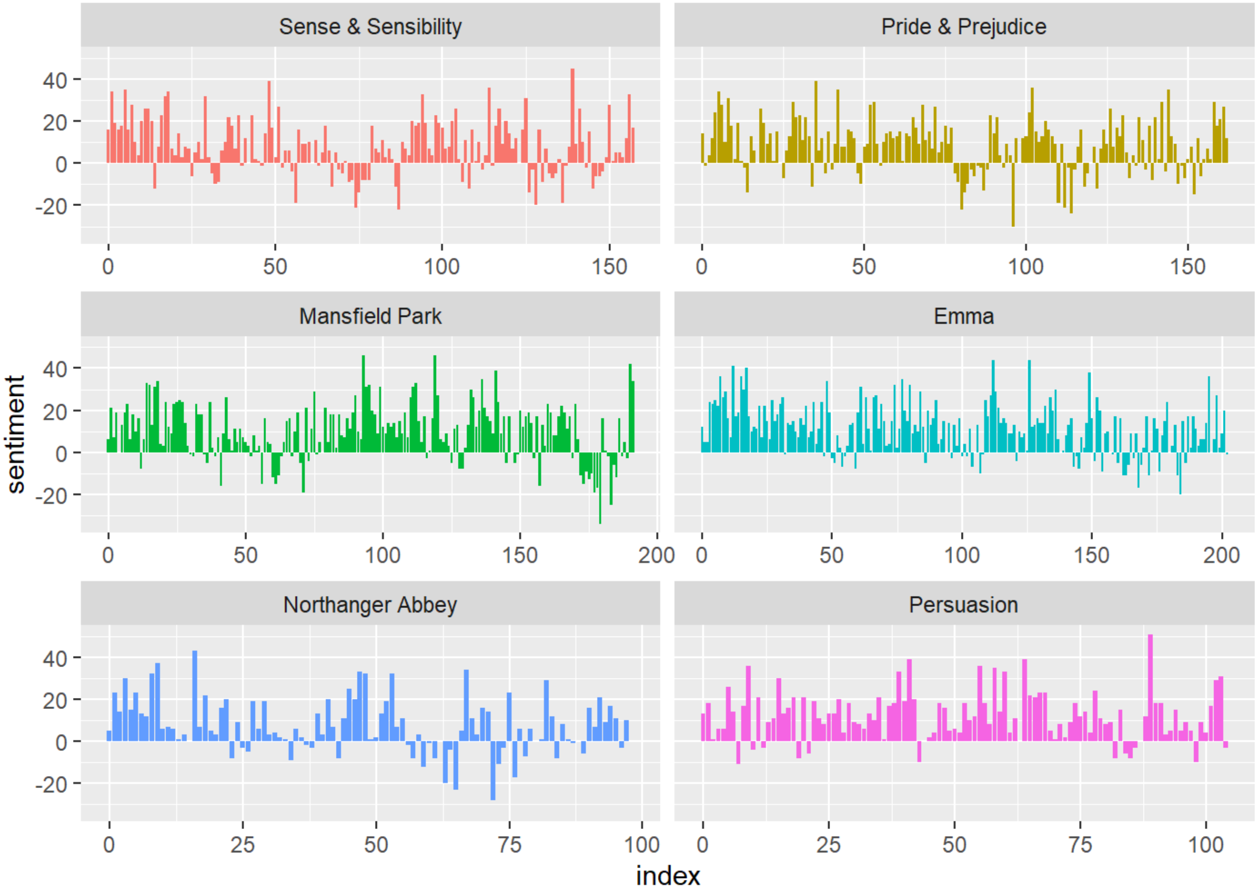
```
## Joining, by = "word"
```

Now we can plot these sentiment scores across the plot trajectory of each novel. Notice that we are plotting against the index on the x-axis that keeps track of narrative time in sections of text.

Hide

```
library(ggplot2)

ggplot(jane_austen_sentiment, aes(index, sentiment, fill = book)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~book, ncol = 2, scales = "free_x")
```



Comparing the three sentiment dictionaries

Hide

```
pride_prejudice <- tidy_books %>%
  filter(book == "Pride & Prejudice")

knitr::kable(head(pride_prejudice))
```

book	linenumber	chapter	word
Pride & Prejudice	1	0	pride
Pride & Prejudice	1	0	and
Pride & Prejudice	1	0	prejudice
Pride & Prejudice	3	0	by

book	linenumber	chapter	word
Pride & Prejudice	3	0	jane
Pride & Prejudice	3	0	austen

Hide

```
afinn <- pride_prejudice %>%
  inner_join(get_sentiments("afinn")) %>%
  group_by(index = linenumber %/% 80) %>%
  summarise(sentiment = sum(value)) %>%
  mutate(method = "AFINN")
```

```
## Joining, by = "word"
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

Hide

```
bing_and_nrc <- bind_rows(
  pride_prejudice %>%
    inner_join(get_sentiments("bing")) %>%
    mutate(method = "Bing et al."),
  pride_prejudice %>%
    inner_join(get_sentiments("nrc") %>%
      filter(sentiment %in% c(
        "positive",
        "negative"
      ))) %>%
    mutate(method = "NRC")
) %>%
  count(method, index = linenumber %/% 80, sentiment) %>%
  spread(sentiment, n, fill = 0) %>%
  mutate(sentiment = positive - negative)
```

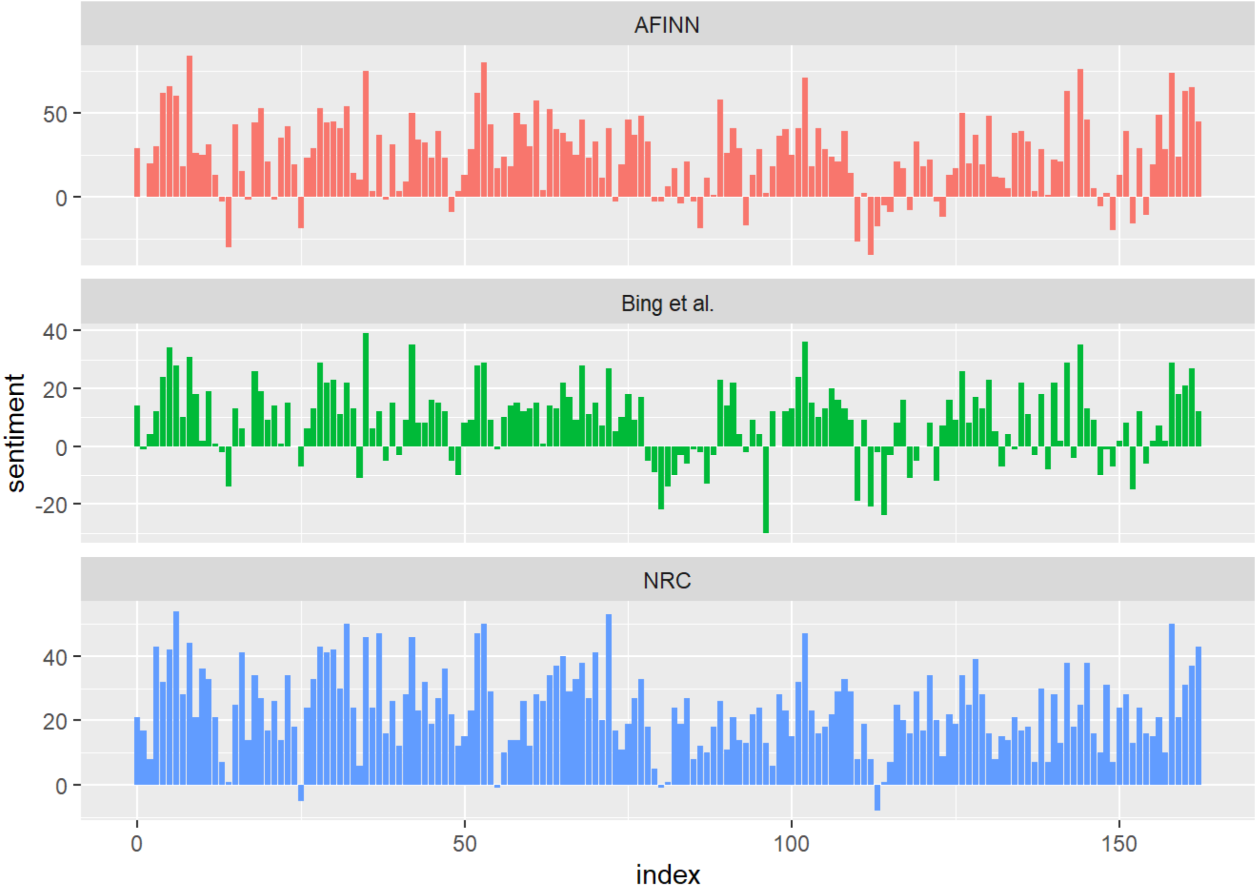
```
## Joining, by = "word"
```

```
## Joining, by = "word"
```

We now have an estimate of the net sentiment (positive - negative) in each chunk of the novel text for each sentiment lexicon. Let’s bind them together and visualize them next:

Hide

```
bind_rows(
  afinn,
  bing_and_nrc
) %>%
  ggplot(aes(index, sentiment, fill = method)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~method, ncol = 1, scales = "free_y")
```



Counting positive and negative words

Hide

```
bing_word_counts <- tidy_books %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort = TRUE) %>%
  ungroup()
```

```
## Joining, by = "word"
```

Hide

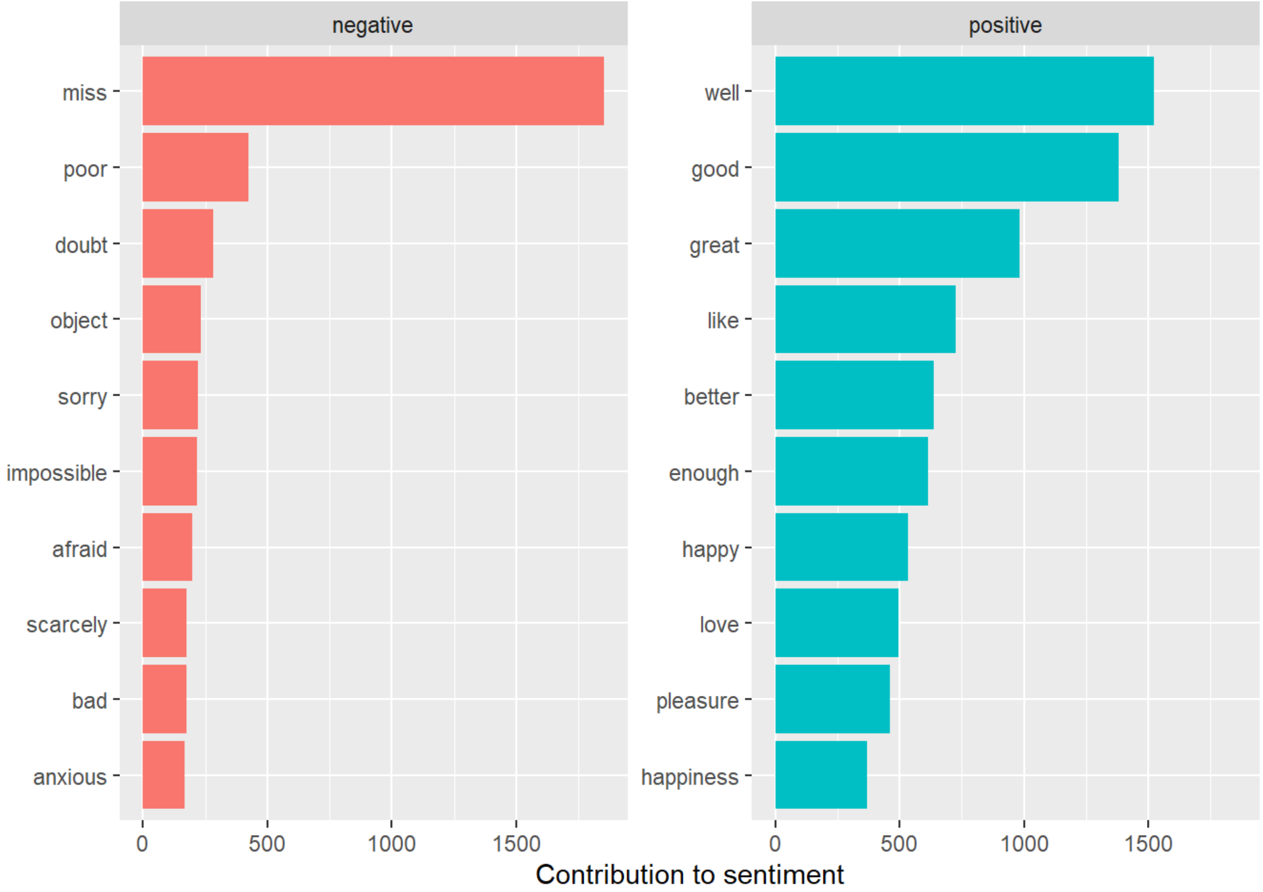
```
bing_word_counts
```

```
## # A tibble: 2,585 x 3
##    word     sentiment     n
##    <chr>    <chr>     <int>
##  1 miss     negative   1855
##  2 well     positive   1523
##  3 good     positive   1380
##  4 great    positive    981
##  5 like     positive    725
##  6 better   positive    639
##  7 enough   positive    613
##  8 happy    positive    534
##  9 love     positive    495
## 10 pleasure positive    462
## # ... with 2,575 more rows
```

Hide

```
bing_word_counts %>%
  group_by(sentiment) %>%
  top_n(10) %>%
  ungroup() %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(word, n, fill = sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free_y") +
  labs(
    y = "Contribution to sentiment",
    x = NULL
  ) +
  coord_flip()
```

```
## Selecting by n
```



This image lets us spot an anomaly in the sentiment analysis; the word “miss” is coded as negative but it is used as a title for young, unmarried women in Jane Austen’s works. We could easily add “miss” to a custom stop-words list using bind_rows().

Wordclouds

Hide

```
library(wordcloud)
```

```
## Warning: package 'wordcloud' was built under R version 4.0.3
```

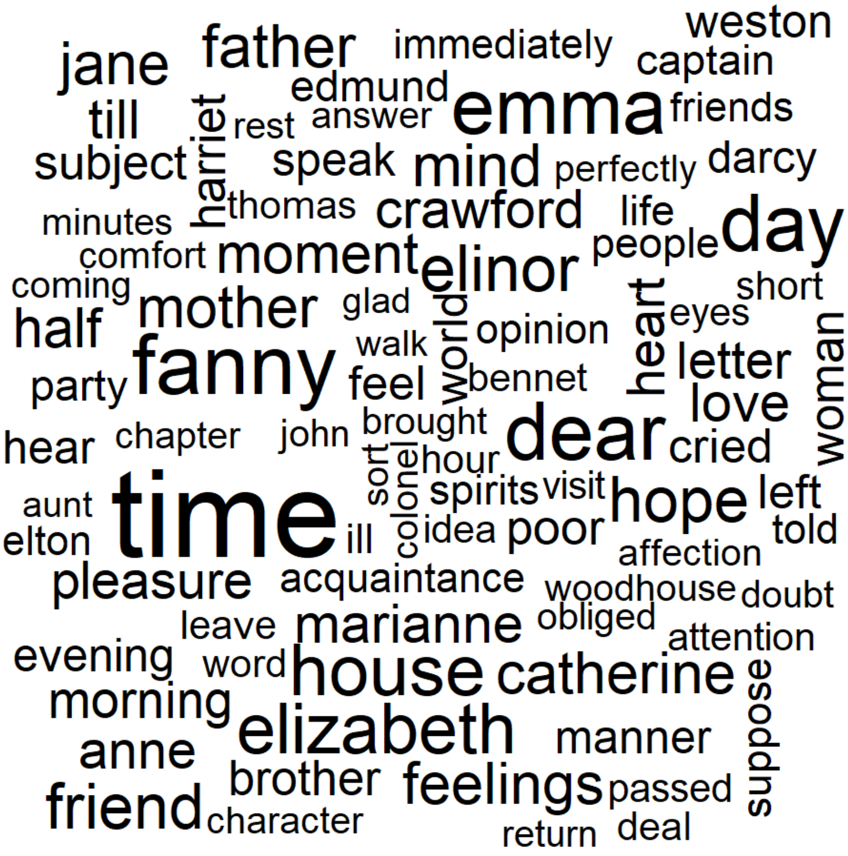
```
## Loading required package: RColorBrewer
```

Hide

```
custom_stop_words <- bind_rows(
  tibble(
    word = c("miss"),
    lexicon = c("custom")
  ),
  stop_words
)

suppressWarnings(tidy_books %>%
  anti_join(custom_stop_words) %>%
  count(word) %>%
  with(wordcloud(word, n, max.words = 100)))
```

```
## Joining, by = "word"
```



Looking at units beyond just words

We can use `unnest_tokens()` to split into tokens using a *regex* pattern. We could use this, for example, to split the text of Jane Austen’s novels into a data frame by chapter.

Hide

```
austen_chapters <- austen_books() %>%
  group_by(book) %>%
  unnest_tokens(chapter, text,
    token = "regex",
    pattern = "Chapter|CHAPTER [\\dIVXLC]"
  ) %>%
  ungroup()

austen_chapters %>%
  group_by(book) %>%
  summarise(chapters = n())
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 6 x 2
##   book                chapters
##   <fct>                  <int>
## 1 Sense & Sensibility       51
## 2 Pride & Prejudice         62
## 3 Mansfield Park            49
## 4 Emma                      56
## 5 Northanger Abbey          32
## 6 Persuasion                25
```

We can use tidy text analysis to ask questions such as what are the most negative chapters in each of Jane Austen's novels? First, let's get the list of negative words from the Bing lexicon. Second, let's make a data frame of how many words are in each chapter so we can normalize for the length of chapters. Then, let's find the number of negative words in each chapter and divide by the total words in each chapter. For each book, which chapter has the highest proportion of negative words?

Hide

```
bingnegative <- get_sentiments("bing") %>%
  filter(sentiment == "negative")

wordcounts <- tidy_books %>%
  group_by(book, chapter) %>%
  summarize(words = n())
```

```
## `summarise()` regrouping output by 'book' (override with `.groups` argument)
```

Hide

```
tidy_books %>%
  semi_join(bingnegative) %>%
  group_by(book, chapter) %>%
  summarize(negativewords = n()) %>%
  left_join(wordcounts, by = c("book", "chapter")) %>%
  mutate(ratio = negativewords / words) %>%
  filter(chapter != 0) %>%
  top_n(1) %>%
  ungroup()
```

```
## Joining, by = "word"
## `summarise()` regrouping output by 'book' (override with `.groups` argument)
```

```
## Selecting by ratio
```

```
## # A tibble: 6 x 5
##   book                chapter negativewords words  ratio
##   <fct>                 <int>         <int> <int>  <dbl>
## 1 Sense & Sensibility      43           161  3405 0.0473
## 2 Pride & Prejudice        34           111  2104 0.0528
## 3 Mansfield Park           46           173  3685 0.0469
## 4 Emma                     15           151  3340 0.0452
## 5 Northanger Abbey         21           149  2982 0.0500
## 6 Persuasion                4            62  1807 0.0343
```

Summary

Sentiment analysis provides a way to understand the attitudes and opinions expressed in texts. In this analysis, we explored how to approach sentiment analysis using tidy data principles; when text data is in a tidy data structure, sentiment analysis can be implemented as an inner join. We can use sentiment analysis to understand how a narrative arc changes throughout its course or what words with emotional and opinion content are important for a particular text.

Self - Exploration

Harry Potter - Sentiment Analysis

We will extend this analysis by using the same techniques explored before and applying them to the Harry Potter books.

I identified this library: Harry Potter Books (https://github.com/bradleyboehmke/harrypotter) which allows us access to the whole Harry Potter texts.

To Install use:

Hide

```
if (packageVersion("devtools") < 1.6) {
  install.packages("devtools")
}

devtools::install_github("bradleyboehmke/harrypotter")
```

Start Analysis

Hide

```
library(harrypotter)
```

The books are stored as character vectors so the first step is to get them as data frames. I got them into separate dataframes, then used `rbind` to make a singular data frame.

Hide

```
# The books are stored as character vectors so
# we need to get them into dataframes

hp1 <- as.data.frame(philosophers_stone) %>%
 mutate(
   book = "1_philosophers_stone",
  chapter = row_number(),
  ) %>%
  unnest_tokens(word, philosophers_stone)

hp2 <- as.data.frame(chamber_of_secrets) %>%
 mutate(
   book = "2_chamber_of_secrets",
  chapter = row_number(),
  ) %>%
  unnest_tokens(word, chamber_of_secrets)

hp3 <- as.data.frame(prisoner_of_azkaban) %>%
 mutate(
   book = "3_prisoner_of_azkaban",
  chapter = row_number(),
  ) %>%
  unnest_tokens(word, prisoner_of_azkaban)

hp4 <- as.data.frame(goblet_of_fire) %>%
 mutate(
   book = "4_goblet_of_fire",
  chapter = row_number(),
  ) %>%
  unnest_tokens(word, goblet_of_fire)

hp5 <- as.data.frame(order_of_the_phoenix) %>%
 mutate(
   book = "5_order_of_the_phoenix",
  chapter = row_number(),
  ) %>%
  unnest_tokens(word, order_of_the_phoenix)

hp6 <- as.data.frame(half_blood_prince) %>%
 mutate(
   book = "6_half_blood_prince",
  chapter = row_number(),
  ) %>%
  unnest_tokens(word, half_blood_prince)

hp7 <- as.data.frame(deathly_hallows) %>%
 mutate(
   book = "7_deathly_hallows",
  chapter = row_number(),
  ) %>%
  unnest_tokens(word, deathly_hallows)

hp_books<-rbind(hp1, hp2, hp3, hp4, hp5, hp6, hp7)
```

** Analyze the sentiments by using bing**

Hide

```
hp_sentiment <- hp_books %>%
  inner_join(get_sentiments("bing")) %>%
  count(book, chapter, sentiment) %>%
  spread(sentiment, n, fill = 0) %>%
  mutate(sentiment = positive - negative)
```

```
## Joining, by = "word"
```

Using the package viridis for styling.

Hide

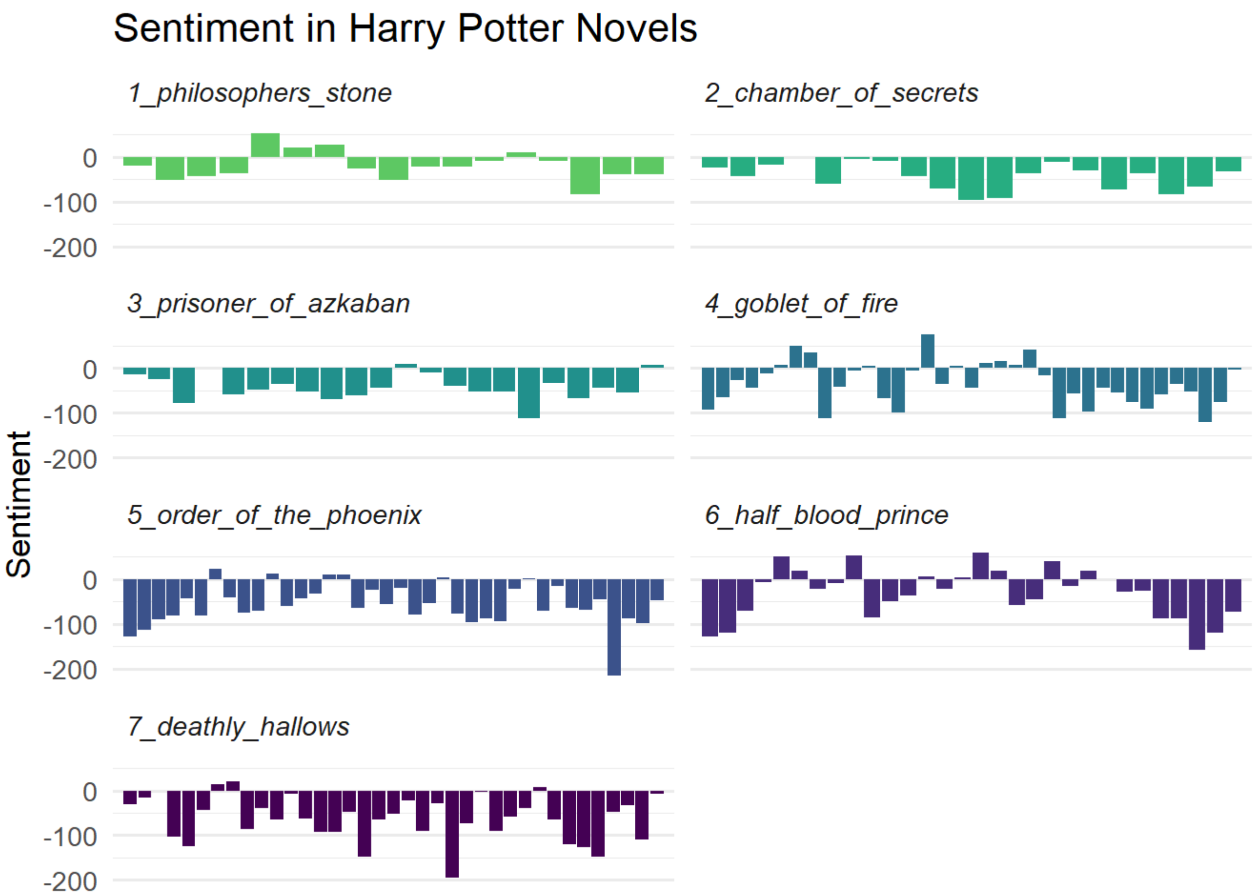
```
library(viridis)
```

```
## Warning: package 'viridis' was built under R version 4.0.3
```

```
## Loading required package: viridisLite
```

Hide

```
ggplot(hp_sentiment, aes(chapter, sentiment, fill = book)) +
        geom_bar(stat = "identity", show.legend = FALSE) +
        facet_wrap(~book, ncol = 2, scales = "free_x") +
        theme_minimal(base_size = 13) +
        labs(title = "Sentiment in Harry Potter Novels",
             y = "Sentiment") +
        scale_fill_viridis(end = 0.75, discrete=TRUE, direction = -1) +
        scale_x_discrete(expand=c(0.02,0)) +
        theme(strip.text=element_text(hjust=0)) +
        theme(strip.text = element_text(face = "italic")) +
        theme(axis.title.x=element_blank()) +
        theme(axis.ticks.x=element_blank()) +
        theme(axis.text.x=element_blank())
```



Based on this graph, it would seem the Harry Potter book overall sentiment is negative.

Finding the most positive chapters in the books

With the Jane Austen novels, which were mostly positive, we tried to take a look at the mostly negative chapters. For the Harry Potter books, we'll try to find the most positive chapters.

Hide

```
bingpositive <- get_sentiments("bing") %>%
        filter(sentiment == "positive")

wordcounts <- hp_books %>%
        group_by(book, chapter) %>%
        summarize(words = n())
```

```
## `summarise()` regrouping output by 'book' (override with `.groups` argument)
```

Hide

```
hp_books %>%
        semi_join(bingpositive) %>%
        group_by(book, chapter) %>%
        summarize(positivewords = n()) %>%
        left_join(wordcounts, by = c("book", "chapter")) %>%
        mutate(ratio = positivewords/words) %>%
        filter(chapter != 0) %>%
        top_n(1)
```

```
## Joining, by = "word"
## `summarise()` regrouping output by 'book' (override with `.groups` argument)
```

```
## Selecting by ratio
```

```
## # A tibble: 7 x 5
## # Groups:   book [7]
##   book                    chapter positivewords words  ratio
##   <chr>                     <int>         <int> <int>  <dbl>
## 1 1_philosophers_stone          5           214  6613 0.0324
## 2 2_chamber_of_secrets         19           265  8568 0.0309
## 3 3_prisoner_of_azkaban        12           156  4797 0.0325
## 4 4_goblet_of_fire              8           201  5860 0.0343
## 5 5_order_of_the_phoenix       15           225  6897 0.0326
## 6 6_half_blood_prince           9           237  5888 0.0403
## 7 7_deathly_hallows            35           180  5008 0.0359
```

BY looking at this table, we see which the most positive chapters of each book are. Chapter 5 on book 1 is when Harry discovers the wonderful world of magic and travels with Hagrid to Diagon Alley. In Deathly Hallows, chapter 35, King's Cross is the calm before the storm. After Voldemort "kills" Harry, he wakes at King's Cross station and has one last meeting with Dumbledore.

WordCloud

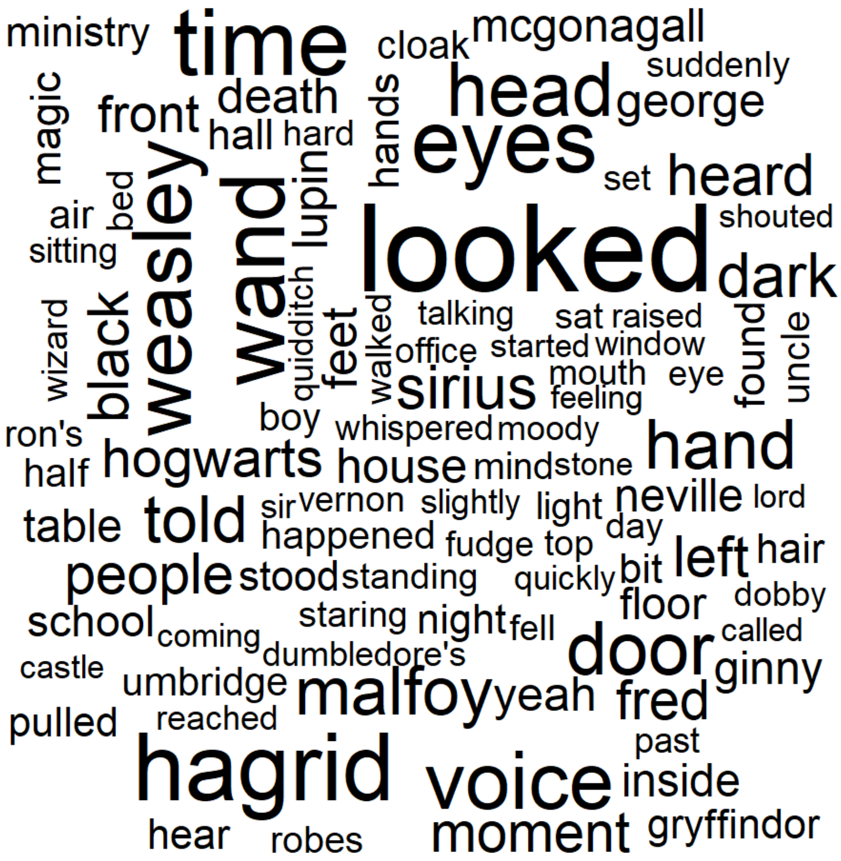
Let's generate a word cloud from Harry Potter's books.

Hide

```
#eliminate the most common names from the wordcloud
custom_stop_words <- bind_rows(
  tibble(
    word = c("harry", "potter", "hermione", "ron", "dumbledore", "voldemort"),
    lexicon = c("custom")
  ),
  stop_words
)

suppressWarnings(hp_books %>%
  anti_join(custom_stop_words) %>%
  count(word) %>%
  with(wordcloud(word, n, max.words = 100)))
```

```
## Joining, by = "word"
```



Conclusion

We can see some of the words like **dark**, **hard**, **fell** and **night** be some of the most common ones. No wonder the overall sentiment of Harry Potter is negative!

Hide

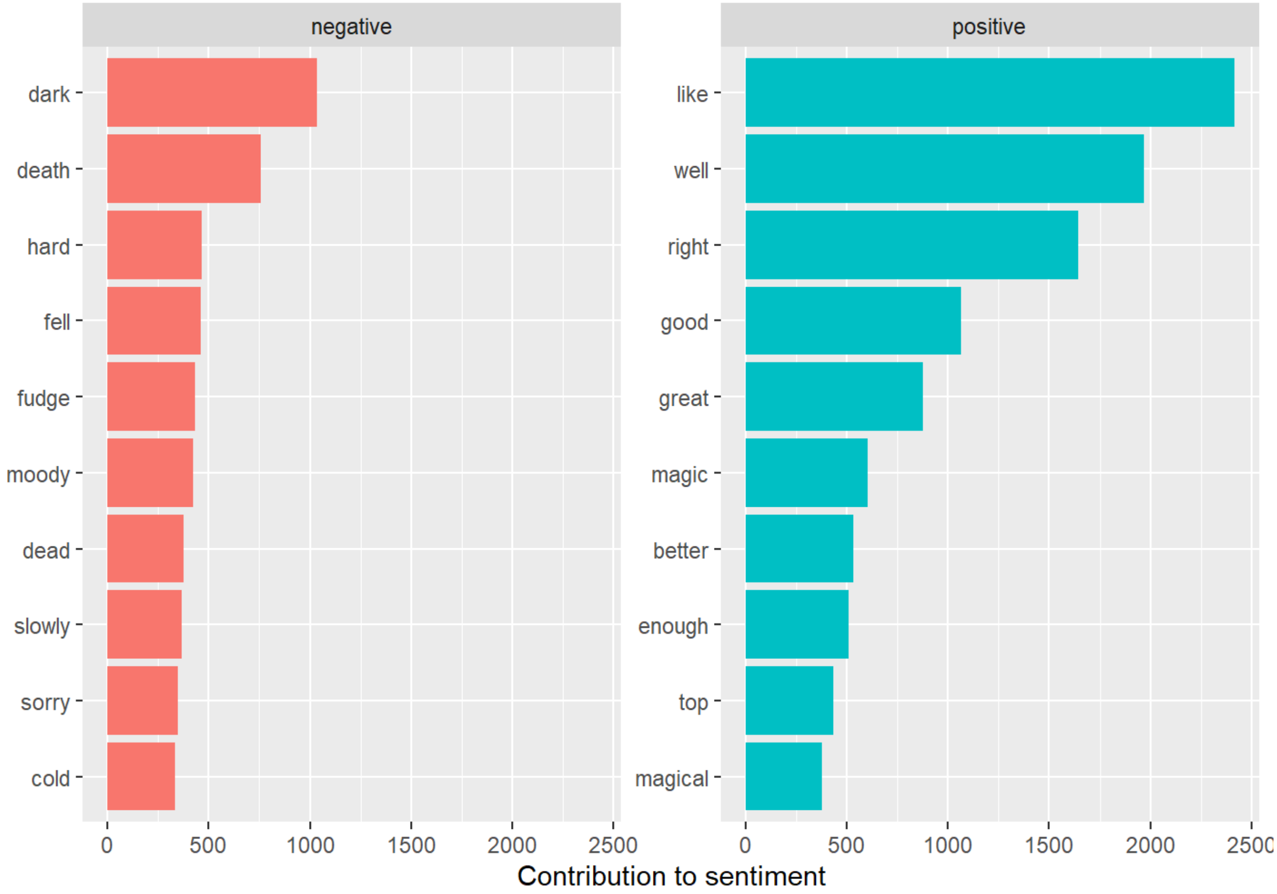
```
hp_word_counts <- hp_books %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort = TRUE) %>%
  ungroup()
```

```
## Joining, by = "word"
```

Hide

```
hp_word_counts %>%
  group_by(sentiment) %>%
  top_n(10) %>%
  ungroup() %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(word, n, fill = sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free_y") +
  labs(
    y = "Contribution to sentiment",
    x = NULL
  ) +
  coord_flip()
```

```
## Selecting by n
```



If we perform a loop at the most used positive and negative words, we see that, even though the overall sentiment of the books is negative, the most used words have a positive charge. This might have something to do with the book’s popularity and sense of uplifting messages.

Even though the Harry Potter series target audience is teenagers and young adults, some of the themes it deals with: prejudice, murder, mistreatment of children, death and loss, can be really hard and dark. It comes as no surprise that the overall sentiment of the books is deemed as negative.

…